Installing Docker

August 29th, 2017

1 Hadoop Docker Image

Configuring a Hadoop cluster on your computer for testing purposes can be a pain. Dealing with inconsistent package versions, having to dive through obscure error messages, and having to wait hours for packages to compile can be frustrating. This makes it hard to get started with Hadoop in the first place. Fortunately, Docker is here to the rescue!

Docker makes it fast and easy to create new environments with the required packages. Docker containers are a layer over Linux containers that makes them easier to manage and distribute. Docker makes it easy to download images that correspond to a specific set of packages, and start them quickly. Docker is cross-platform, and works on Mac, Windows, and Linux.

In this tutorial, well cover the basics of Docker, how to install it, and how to leverage Docker containers to quickly get started with Hadoop on your own machine using a docker container. Later in the course, you may be required to download other container that support other Big Data tools.

1.1 Getting started with Docker

The first step is installing Docker. Theres a graphical installer for Windows and Mac that makes this easy. Here are the instructions for each OS:

1. Windows : `https://docs.docker.com/docker-for-windows/`
2. MacOS `https://docs.docker.com/docker-for-mac/`

As part of this installation process, you'll need to use a shell prompt. The shell prompt, also called the terminal or the command line, is a way to run commands on your machine from a text interface instead of graphically.

On Windows, the shell is called Command Prompt, while on Mac and Linux it is called Terminal.

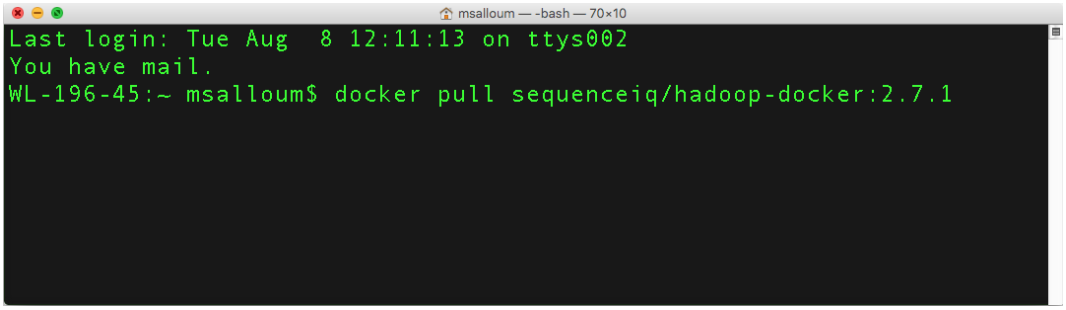
Youll need to use this same shell prompt whenever the rest of this post mentions having to run a Docker command or type a specific command.

1.2 Downloading the Image

Once Docker is downloaded, the next step is to download the image/configuration you want. Open the terminal window if using Mac or the command prompt if you are using Windows. Then type:

docker pull sequenceiq/hadoop-docker:2.7.1

This pull downloads the image along with all the libraries and tools. This command is issued only the first time to download the image. Next time, you can jump to step 1.2.2 directly without doing a pull.



1.2.1 Make a folder

Make a folder on your local machine that will correspond to where you want the code stored. This folder will contain all of your work, and will persist on your local machine, even if you terminate the docker container. For example, I made a folder under the folder: /Users/msalloum/cs181.

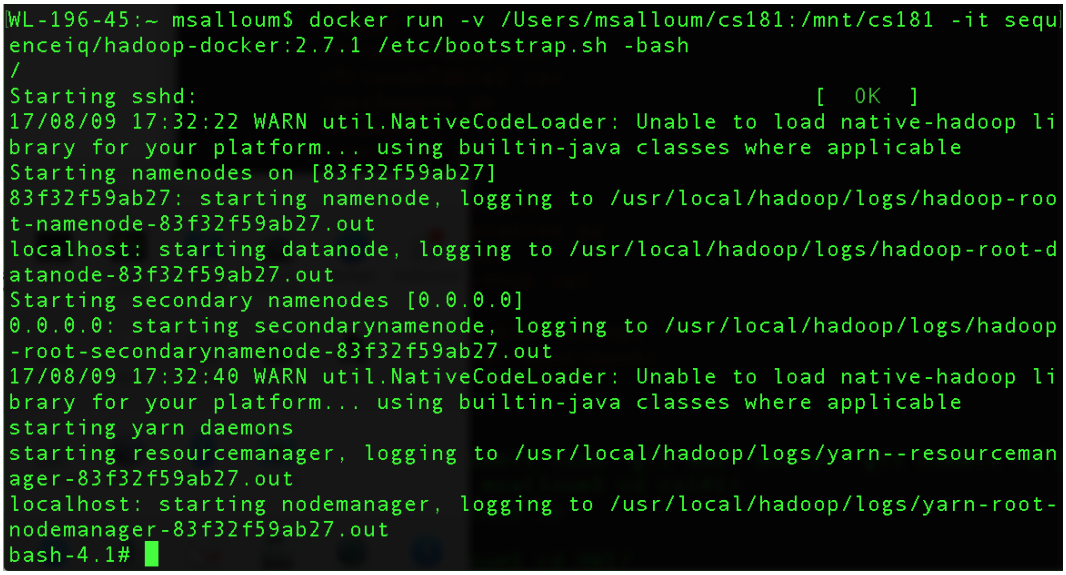
1.2.2 Running the image

Once you download the image, you can run it using docker run. We need to pass in one specific argument which specifies which directory on the local machine contains our code, so that the Docker image will have access to our code.

The full command looks like:

docker run -v /Users/msalloum/cs181:/mnt/cs181 -it sequenceiq/hadoop-docker:2.7.1 /etc/bootstrap.sh -bash

You should change /Users/msalloum/cs181 to whatever folder you created to store your code.



If successful, it will eventually enter a "bash" script mode, where we can run MapReduce example.

1.3 Running an example

You can run an example MapReduce job to verify everything works correctly:

cd /usr/local/hadoop

bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar grep input output 'dfs[a-z.]+'

```
bash-4.1# pwd
/usr/local/hadoop
bash-4.1# bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar grep i
nput output 'dfs[a-z.]+'
17/08/09 17:36:06 WARN util.NativeCodeLoader: Unable to load native-hadoop library for you
r platform... using builtin-java classes where applicable
17/08/09 17:36:07 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
17/08/09 17:36:09 INFO input.FileInputFormat: Total input paths to process : 31
17/08/09 17:36:10 INFO mapreduce.JobSubmitter: number of splits:31
17/08/09 17:36:10 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_150231436400
2_0001
17/08/09 17:36:11 INFO impl.YarnClientImpl: Submitted application application_150231436400
2_0001
17/08/09 17:36:11 INFO mapreduce.Job: The url to track the job: http://83f32f59ab27:8088/p
roxy/application_1502314364002_0001/
17/08/09 17:36:11 INFO mapreduce.Job: Running job: job_1502314364002_0001
17/08/09 17:36:21 INFO mapreduce.Job: Job job_1502314364002_0001 running in uber mode : fa
lse
17/08/09 17:36:21 INFO mapreduce.Job:  map 0% reduce 0%
17/08/09 17:36:49 INFO mapreduce.Job:  map 19% reduce 0%
17/08/09 17:37:31 INFO mapreduce.Job:  map 35% reduce 0%
17/08/09 17:37:35 INFO mapreduce.Job:  map 35% reduce 12%
17/08/09 17:37:46 INFO mapreduce.Job:  map 45% reduce 12%
17/08/09 17:37:47 INFO mapreduce.Job:  map 52% reduce 15%
17/08/09 17:37:50 INFO mapreduce.Job:  map 52% reduce 17%
17/08/09 17:38:00 INFO mapreduce.Job:  map 61% reduce 17%
17/08/09 17:38:01 INFO mapreduce.Job:  map 68% reduce 17%
17/08/09 17:38:02 INFO mapreduce.Job:  map 68% reduce 22%
17/08/09 17:38:05 INFO mapreduce.Job:  map 68% reduce 23%
17/08/09 17:38:14 INFO mapreduce.Job:  map 77% reduce 23%
```

Be sure to first 'cd' into the '/usr/local/hadoop/'. From there to run a hadoop job, simple type 'bin/hadoop'. Its probably easier to set an enviromental variable to this path so you don't have to type it out every time. To do this, simple do:

export HADOOP_HOME=/usr/local/hadoop

then to run a hadoop job, type '$HADOOP_HOME/bin/hadoop'